**Program 1:Resume Screening**

import re

file\_path = "./res1.txt"

print()

with open(file\_path, "r", encoding="utf-8") as file:

  resume\_text = file.read()

words = resume\_text.splitlines()

name = words[0] if words else "No name found"

email\_pattern = r'\b[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,7}\b'

email = re.search(email\_pattern, resume\_text)

email = email.group(0) if email else "No email found"

phone\_pattern = r'\b\d{3}[-.\s]??\d{3}[-.\s]??\d{4}\b'

phone = re.search(phone\_pattern, resume\_text)

phone = phone.group(0) if phone else "No phone number found"

if name != "No name found" and email != "No email found" and phone != "No phone number found":

  decision = "Sorry, you are not selected for the interview."

else:

  decision = "Congratulations, you are selected for the interview!"

print(f"Name: {name}")

print(f"Email: {email}")

print(f"Phone: {phone}")

print(f"Decision: {decision}",)

**Input:**

Phills

Phone: 123-456-7890

Objective:

Highly skilled Software Engineer with 5+ years of experience in web development and software design.

Proficient in Python, Java, and JavaScript. Looking for opportunities to work on challenging projects.

Work Experience:

Software Engineer | XYZ Technologies | June 2020 - Present

- Developed web applications using Python and JavaScript.

- Worked closely with the design team to create user-friendly interfaces.

Education:

Bachelor of Science in Computer Science | ABC University | 2016 - 2020

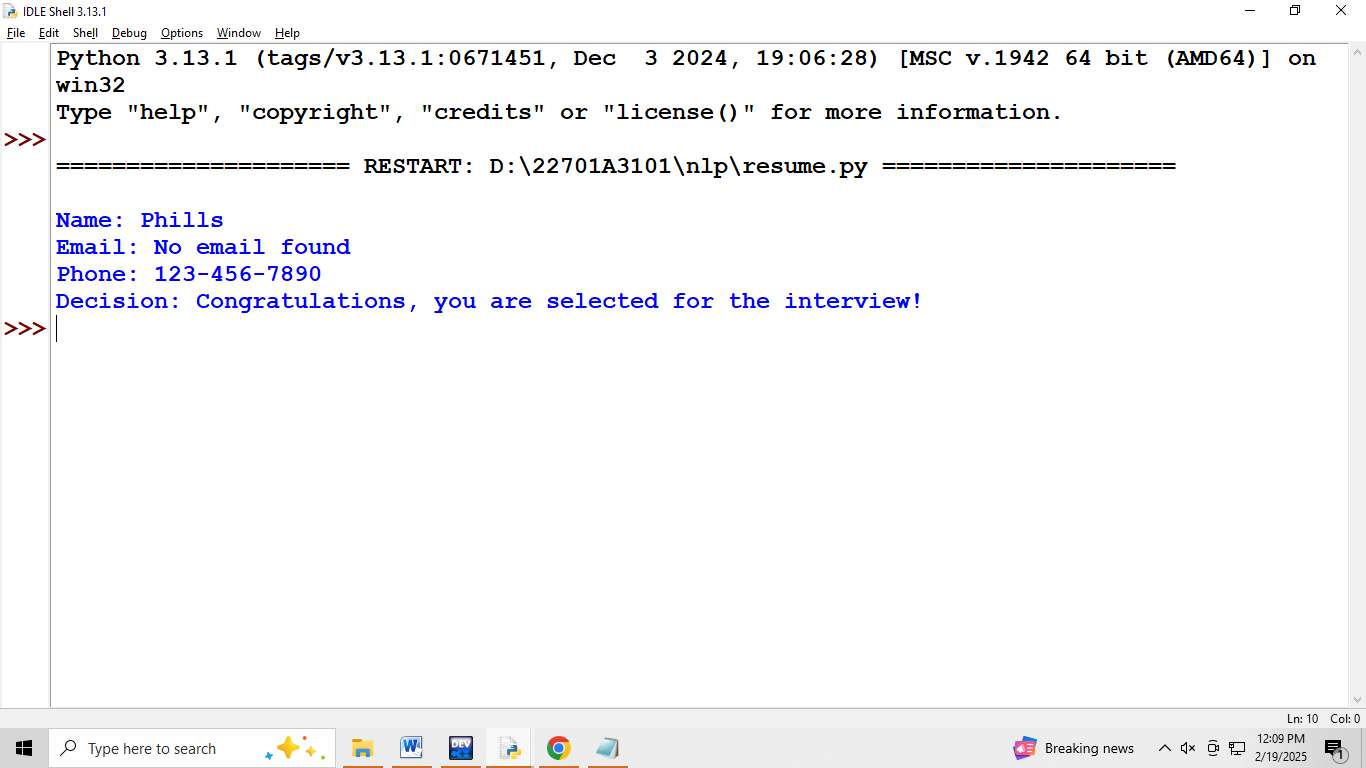
Skills:

- Programming Languages: Python, Java, JavaScript

- Frameworks: Django, Flask, React

- Tools: Git, Docker, Jenkins

**Output:**



**Program 2:Chatbot**

import nltk

from nltk.chat.util import Chat, reflections

pairs = [

    [

        r"my name is (.\*)",

        ["Hello %1, How are you today ?",]

    ],

     [

        r"(.\*)testing you",

        ["i heard sureka mam one of coolest professor in entire campus",]

    ],

    [

        r"i am (.\*)",

        ["so what!",]

    ],

    [

        r"hi|hey|hello",

        ["Hello", "Hey there",]

    ],

    [

        r"what is your name ?",

        ["I am a bot created by Abu. you can call me crazy!",]

    ],

    [

        r"how are you ?",

        ["I'm doing goodnHow about You ?",]

    ],

    [

        r"sorry (.\*)",

        ["Its alright","Its OK, never mind",]

    ],

    [

        r"I am fine",

        ["Great to hear that, How can I help you?",]

    ],

    [

        r"i'm (.\*) doing good",

        ["Nice to hear that","How can I help you?:)",]

    ],

    [

        r"(.\*) age?",

        ["I'm a computer program dudenSeriously you are asking me this?",]

    ],

    [

        r"what (.\*) want ?",

        ["Make me an offer I can't refuse",]

    ],

    [

        r"(.\*) created ?",

        ["Raghav created me using Python's NLTK library ","top secret ;)",]

    ],

    [

        r"(.\*) (location|city) ?",

        ['Indore, Madhya Pradesh',]

    ],

    [

        r"how is weather in (.\*)?",

        ["Weather in %1 is awesome like always","Too hot man here in %1","Too cold man here in %1","Never even heard about %1"]

    ],

    [

        r"i work in (.\*)?",

        ["%1 is an Amazing company, I have heard about it. But they are in huge loss these days.",]

    ],

    [

        r"(.\*)raining in (.\*)",

        ["No rain since last week here in %2","Damn its raining too much here in %2"]

    ],

    [

        r"how (.\*) health(.\*)",

        ["I'm a computer program, so I'm always healthy ",]

    ],

    [

        r"(.\*) (sports|game) ?",

        ["I'm a very big fan of Football",]

    ],

    [

        r"who (.\*) sportsperson ?",

        ["Messy","Ronaldo","Roony"]

    ],

    [

        r"who (.\*) (moviestar|actor)?",

        ["Brad Pitt"]

    ],

    [

        r"i am looking for online guides and courses to learn data science, can you suggest?",

        ["Crazy\_Tech has many great articles with each step explanation along with code, you can explore"]

    ],

    [

        r"quit",

        ["Bye take care. See you soon :) ","It was nice talking to you. See you soon :)"]

    ],

]

def chat():

    print("Hi! I am a chatbot created by Abu for your service")

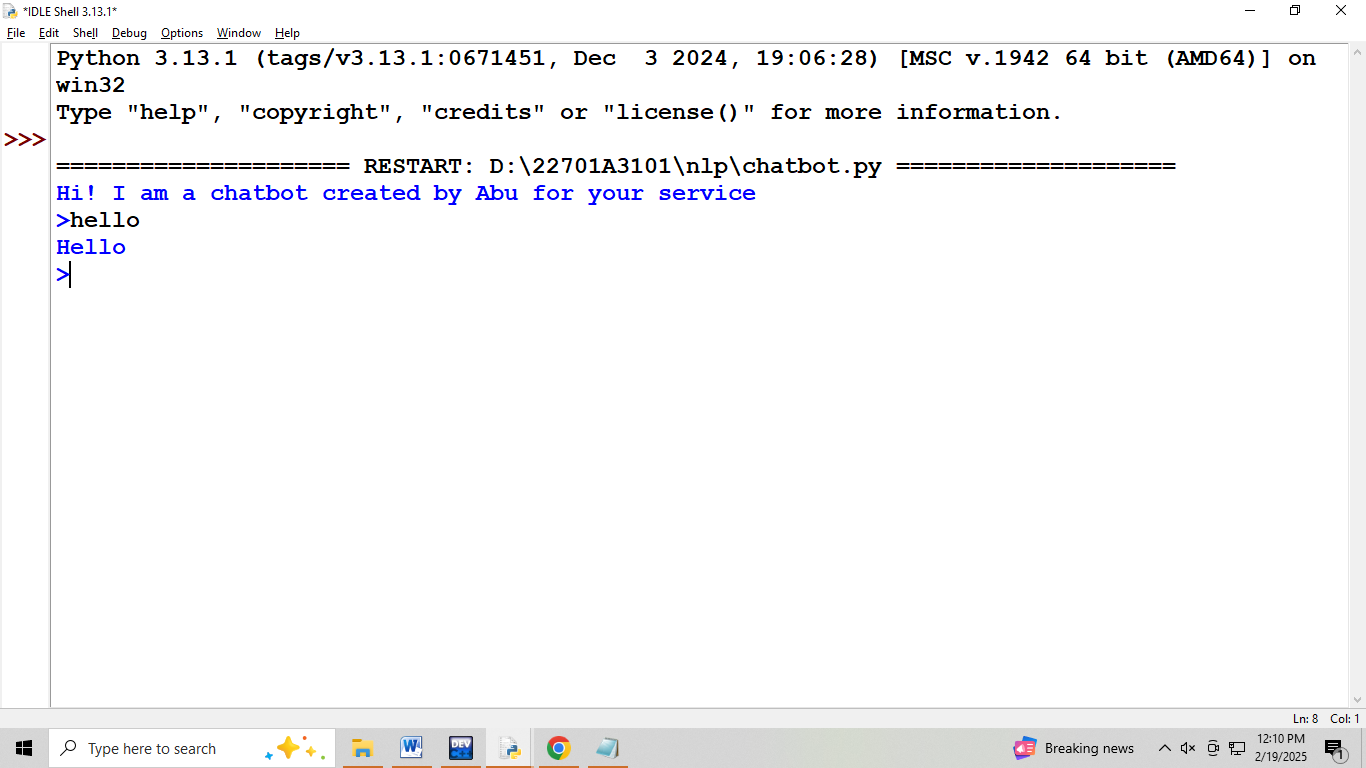
    chat = Chat(pairs, reflections)

    chat.converse()

if \_\_name\_\_ == "\_\_main\_\_":

    chat()

**Output:**



**Program 3:Named Entity**

import re

def extract\_entities(text):

    entities=[]

    date\_pattern=r'\d{1,2}^d{1,2}^d{2,4}'

    number\_pattern=r'\d+'

    name\_pattern=r"[A-z][a-z]+"

    dates=re.findall(date\_pattern,text)

    entities.extend([('DATE',date) for date in dates])

    numbers=re.findall(number\_pattern,text)

    entities.extend([('NUMBER',number) for number in numbers])

    names=re.findall(name\_pattern,text)

    entities.extend([('NAME',name) for name in names])

    return entities

if \_\_name\_\_=="\_\_main\_\_":

    text="on 28/01/2025.,siva received $500 from madhu"

    entities=extract\_entities(text)

    if entities:

        print("entities found")

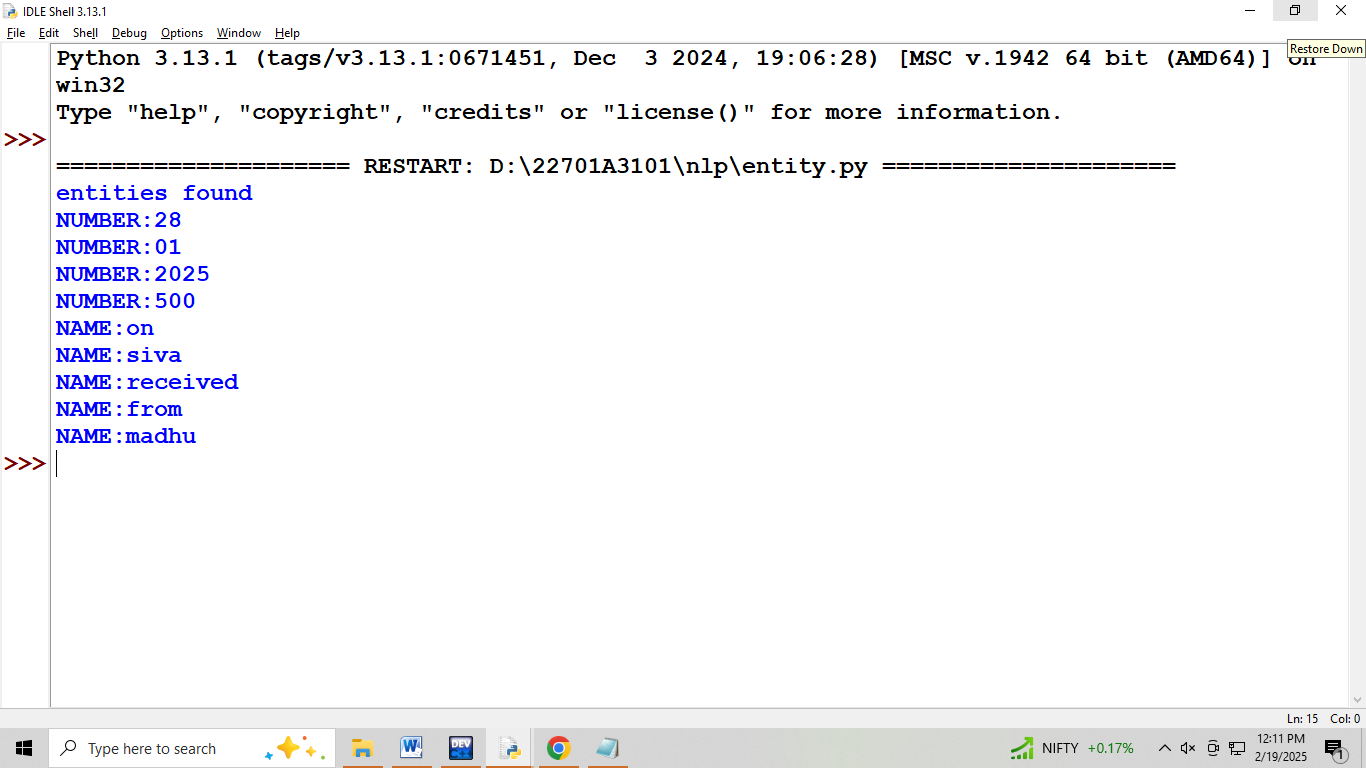
        for entity\_type,entity\_value in entities:

            print(f"{entity\_type}:{entity\_value}")

    else:

        print("no entities found in the text")

**Output:**



**Program 4:Parts of speech**

import nltk

from nltk.tokenize import word\_tokenize

from nltk import pos\_tag

# Download required NLTK data (only needed once)

def get\_pos\_tags(text):

    words = word\_tokenize(text)  # Tokenize text into words

    pos\_tags = pos\_tag(words)  # Get part of speech for each word

    return pos\_tags

# Example usage:

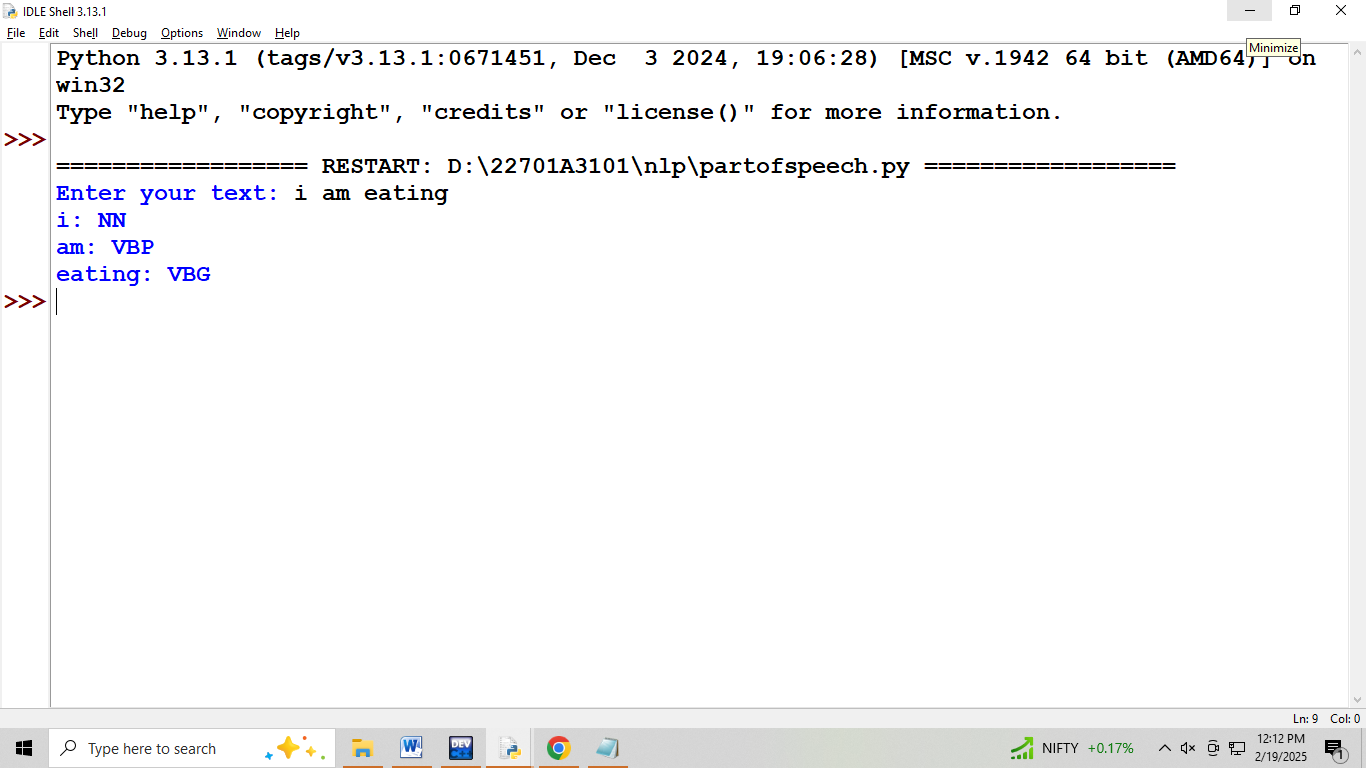
text = input("Enter your text: ")

pos\_result = get\_pos\_tags(text)

# Print result

for word, pos in pos\_result:

    print(f"{word}: {pos}")

**Output:** ****

**Program 5:Sentimental Analysis**

import cv2

import numpy as np

def detect\_emotion\_by\_color(image\_path):

    image = cv2.imread(image\_path)

    if image is None:

        return "Error: Unable to load image. Check the file path."

    # Convert image to HSV (Hue, Saturation, Value)

    hsv = cv2.cvtColor(image, cv2.COLOR\_BGR2HSV)

    # Calculate the average hue (color tone)

    avg\_hue = np.mean(hsv[:, :, 0])

    # Emotion mapping based on color psychology

    if avg\_hue < 20 or avg\_hue > 160:

        emotion = "Angry (Red tones detected)"

    elif 20 <= avg\_hue < 40:

        emotion = "Happy (Yellow tones detected)"

    elif 40 <= avg\_hue < 75:

        emotion = "Relaxed (Green tones detected)"

    elif 75 <= avg\_hue < 130:

        emotion = "Sad (Blue tones detected)"

    else:

        emotion = "Neutral (Balanced colors detected)"

    return f"Detected Emotion: {emotion}"

# Get image path from user

for i in range(0,10):

   ip=input("Enter the image file path (e.g., ./angry.jpg): ")

   if(ip=='exit'):

       break

   else:

       image\_path =ip.strip()

       print(detect\_emotion\_by\_color(image\_path))

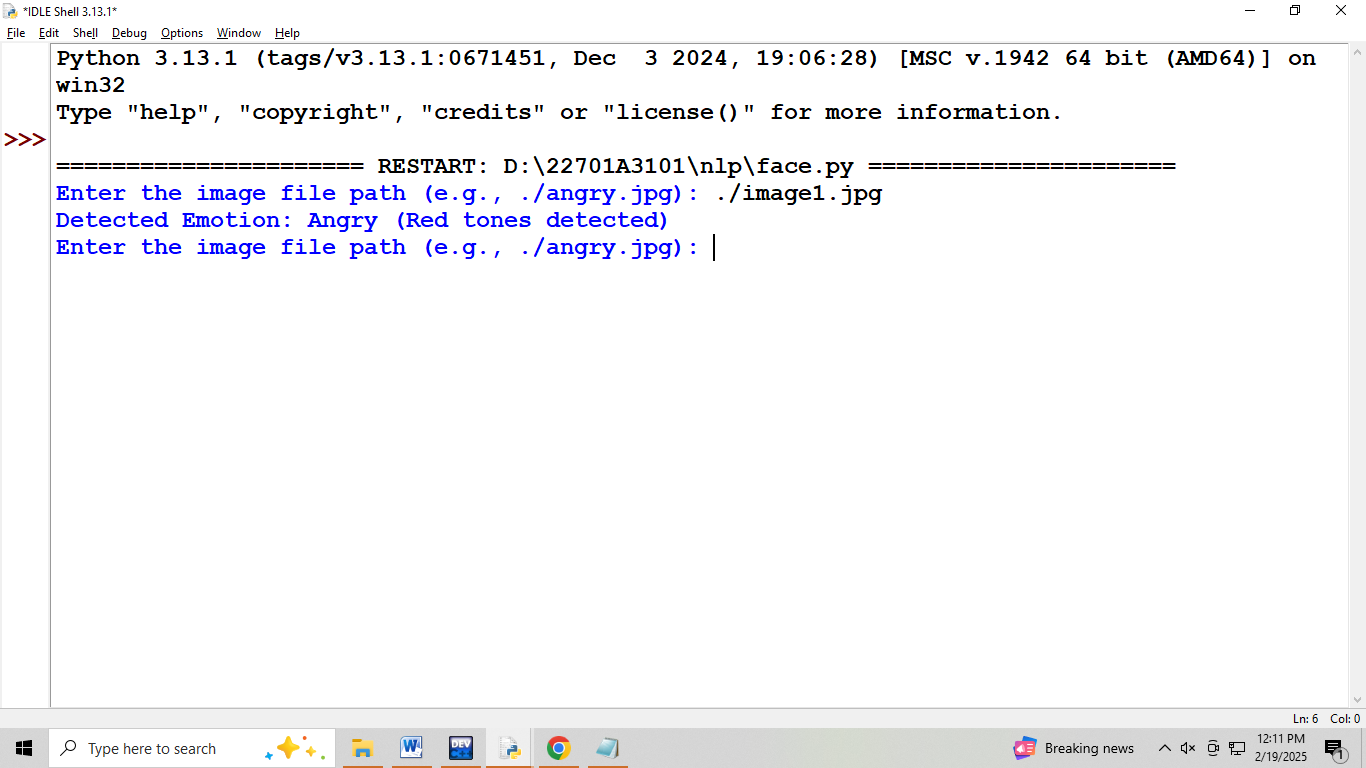
**Input:**

****

****

****

**Output:**

****